A New Obstacle Avoidance Method for Service Robots in Indoor Environments

Widodo Budiharto[1], Ari Santoso[2], Djoko Purwanto[2] & Achmad Jazidie[2]

[1]Bina Nusantara University, Jl. K. H. Syahdan No. 9,
Kemanggisan/Palmerah, Jakarta Barat 11480, Indonesia
[2]Electrical Engineering Department, Faculty of Industrial Technology
Institute of Technology Sepuluh Nopember Surabaya (ITS)
Kampus ITS, Keputih, Surabaya, Jawa Timur 60111, Indonesia
Email: wbudiharto@binus.edu

Abstract. The objective of this paper is to propose an obstacle avoidance method for service robots in indoor environments using vision and ultrasonic sensors. For this research, the service robot was programmed to deliver a drinking cup from a specified starting point to the recognized customer. We have developed three main modules: one for face recognition, one for obstacle detection, and one for avoidance maneuvering. The obstacle avoidance system is based on an edge-detection method using information from the landmark and planned-path generation. Speed, direction and distance of the moving obstacle are measured using vision and distance sensors in order for the robot to make an avoidance maneuver. Algorithms for obstacle avoidance are proposed and a new geometric model is introduced for making good avoidance maneuvers. The main aim of this research is to provide a complete mechanism for obstacle avoidance by vision-based service robots, where common obstacle avoidance methods, such as PVM, do not provide such a feature. We present the results of an experiment with a service robot in which the proposed method was implemented, after which its performance is evaluated.

Keywords: *moving obstacle; obstacle avoidance; robot vision; service robot.*

1 Introduction

Service robot development is an emerging technology that will be popular in the future. A typical application is a service robot that can recognize people and detect obstacles, indoors or outdoors, and accomplish a specific task given by the user. Obstacle avoidance for service robots in indoor environments under varying illumination conditions is complex and challenging. The service robot has to maneuver appropriately in order to avoid static or moving obstacles and reach its target in an optimal manner. This necessitates robust algorithms for collision avoidance, path planning and autonomous action by the service robot [1-4].

Received September 30th, 2010, Revised April 27th, 2011, 2nd Revision August 1st, 2011, 3rd Revision February 16th, 2012, 4th Revision May 1st, 2012, Accepted for publication May 2nd, 2012.

In recent years, many techniques have been developed to carry out obstacle avoidance efficiently by using recent sensor data [5,6]. General obstacle avoidance can be accomplished with different sensors, such as vision, sonar and laser sensors. The potential field method (PFM) and virtual force histogram (VFH) for obstacle avoidance have gained increasing popularity in the field of mobile robots. PFM is based on attractive and repulsive potential fields that guide the robot to reach its target, or enable it to avoid obstacles. PFM has inherent limitations, such as no passage between closely spaced obstacles, oscillations in the presence of obstacles and in narrow passages, and standstill when attractive and repulsive forces are equally strong. VFH is a fast obstacle avoidance method based on a polar histogram of obstacles that provides directions for safe travel. VFH also has shortcomings: the results are sensitive to thresholds and it requires expensive hardware because it uses a histogram grid world model that has to be updated constantly by rapidly firing 24 sensors during motion [7,8]. In the case of vision-based service robots, PFM and VFH do not provide methods for obstacle avoidance or maneuvering.

Another main challenge for service robot development is to detect moving obstacles accurately in the landmark, especially using a vision sensor (camera). The most notable algorithms for detecting moving objects from a moving platform using only a vision sensor can be grouped into two distinct classes, those using optical flow and those using qualitative estimates of motion. A disadvantage of both methods is the difficulty to compute the optical flow with an acceptably low level of noise [9], and also the moving entities must be detected and their future position needs to be predicted over a finite time [10-13].

Literature study by the authors showed that in many research projects, such as [1] and [2], the task of the service robot is setting and clearing tables in a controlled environment. However, no flexible obstacle avoidance method for vision-based service robots in indoor environments is available. In previous papers, we have already proposed a novel method for static and moving obstacle avoidance for vision-based service robots, using Bayesian filtering and an ANFIS controller, but this method uses predefined maneuvering [14,15]. Obstacle avoidance methods based only on ultrasonic sensors must account for the sensors' shortcomings, such as inaccuracies, crosstalk, and spurious readings. Therefore, we have considered a combination of vision and ultrasonic sensors. Vision and ultrasonic sensors are important for the face recognition system, the tracking system and distance measurement. This paper introduces a new obstacle avoidance method for service robots in indoor environments using one single camera. A complete mechanism is proposed for vision-based robots that are able to avoid obstacles autonomously, which is presented in section 2 and 3. Finally, the results of the implementation of the proposed method and

algorithms in a service robot, Srikandi II, are presented in section 4. The discussion of the results in section 5 includes a comparison with the PFM and VFH methods. Section 6 contains the conclusion.

2 Introduction to Vision-Based Service Robots

2.1 Principles and Kinematics of Vision-Based Service Robots

The robot used in this research is a mobile robot equipped with two actuator wheels and is considered as a system subject to nonholonomic constraints. Consider a basic configuration, consisting of an autonomous wheeled mobile robot, its target and a moving obstacle, as shown in Figure 1. When a robot inside an indoor environment moves from its start position to the target, there is the possibility of a moving obstacle hitting the robot. The robot needs relevant information, such as the distance to the obstacle d_O, the velocity of the obstacle v_O, the orientation of the robot θ_R, and the velocity of the robot v_R, in order to be able to take action to avoid the obstacle and make a maneuver with a certain direction angle.

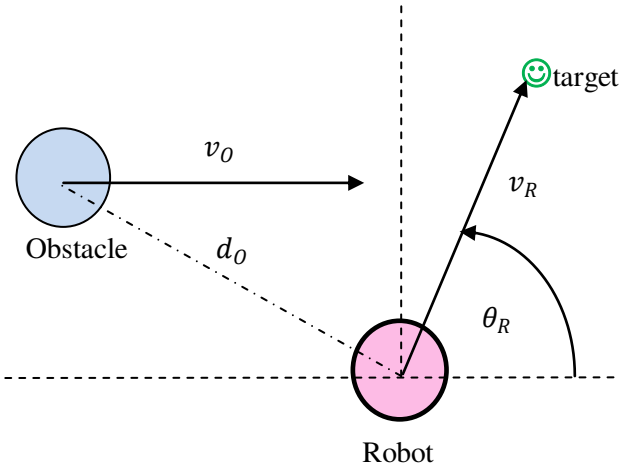


Figure 1 Configuration of mobile robot, moving obstacle and target.

Based on the configuration in Figure 1, we propose a model of a vision-based service robot using a single camera, and a moving obstacle as shown in Figure 2. The camera is an important sensor if we want the robot to detect specific objects (such as a face, a small object, a shape, etc.) that cannot be detected by other sensors, such as an ultrasonic sensor. A camera as a vision sensor has view angle limitations in capturing objects, so we define θ_{Cam} as the maximum angle at which the moving obstacle can be detected by the camera used in this research.

We identify $\theta_{O,}\theta_{OR,}\theta_{Cam},\theta_R,v_R$ and v_O as important properties to calculate whether the robot will collide with the moving obstacle or not. Based on Figure 2, we define the angle between the moving obstacle and the robot θ_{OR} and the orientation of the robot θ_R as:

$$\theta_{OR} = 180^{\circ} - (\theta_R + \theta_{cam}) \tag{1}$$

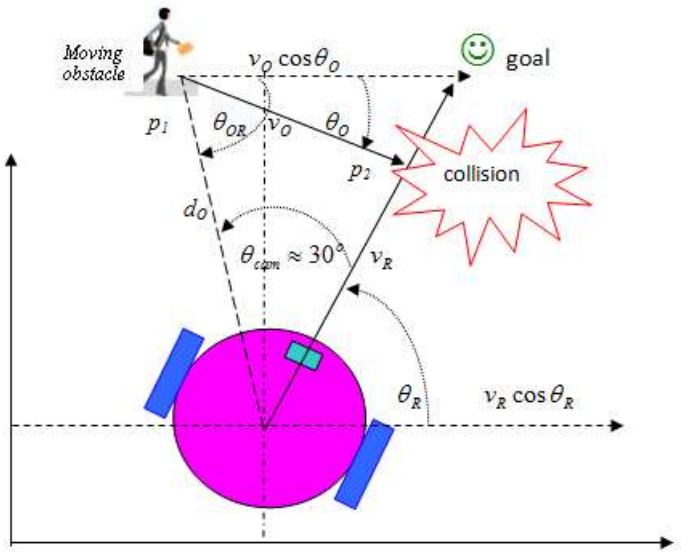


Figure 2 Proposed Cartesian representation of a robot with one camera, moving obstacle, and target.

Theoretically, to measure the speed of an obstacle, the robot should continuously track the obstacle using a vision sensor such as a camera. We propose the model to calculate v_O moving with an angle θ_O detected by the camera, while at the same time the robot moves with the speed v_R to the target with an angle θ_R. In our method we need two points of tracked images (p_1 and p_2) with interval time t, after which the difference in pixel positions is obtained. Based on Figure 2, the equation to estimate v_O if the obstacle and the robot are moving, and the obstacle appears from the left side of the robot, is:

$$v_O \cos\theta_O = \frac{|p_2 - p_1|\, s}{t} + v_R \cos\theta_R \tag{2}$$

Finally, we can simplify Eq. (2) and add constraints, such as:

$$v_O = \begin{cases} \dfrac{|p_2 - p_1|\, s}{t \cos\theta_O} + \dfrac{v_R \cos\theta_R}{\cos\theta_O} & \text{if } 0° <= \theta_O <= 60° \\ v_O \max & \text{if } 60° < \theta_O <= 90° \end{cases} \quad (3)$$

From Eq. (3), if θ_O lies between $60° < \theta_O <= 90°$, it is difficult to estimate the velocity of the obstacle, especially when $\theta_O = 90°$ (direct approach of the robot), therefore, in this situation we assume v_O to be the maximum velocity allowed (we define v_O max=80 cm/s). If the calculated result of $v_O >= 80$ cm/s, then we set v_O to the maximum velocity allowed. p_1 and p_2 are the positions of the moving obstacle in pixels; the scaling factor is in cm/pixel. In principle, Eq. (3) can also be used for a moving obstacle that appears from the right side of the robot. We also propose a mechanism to predict a collision, using time *t*. A collision occurs when the robot hits a person or a moving obstacle that moves with a specific orientation, as shown in Figure 2. We assume that the moving obstacle has the intelligence not to hit the robot when it is standing still. T is used as a time threshold, and can be calculated using the following formula:

$$t = \frac{d_O \sin\theta_{OR}}{(v_R \sin\theta_R + v_O \sin\theta_O)} \quad (4)$$

where: if $t <= T$ then the robot stops

if $t > T$ then the robot moves forward

2.2 Architecture of Vision-Based Service Robot

Our system consists of a camera, which obtains a frontal view for object tracking, face recognition and static obstacle detection, a compass and three ultrasonic sensors for distance measurement. Figure 3 shown below is a prototype of Srikandi II.

Figure 3 Vision-based service robot Srikandi II, developed and used in this research, equipped with a 4 DOF arm robot. The camera has a 640x480 pixel resolution. A compass sensor and three ultrasonic sensors are used as additional sensors for distance measurement, and a laptop is used for image processing [15].

There is an interface program on the laptop for coordinating the robot controller. A 4 DOF arm robot with predefined motion is used to hand over the cup to the customer when the robot arrives at the target position. One master controller is used as coordinating actuator and for communication with the laptop, and another controller is used for distance measurement. Figure 4 is the architecture of service robot Srikandi II. Because this robot needs to recognize and track people, many integrated supporting functions were developed, such as a face recognition system, static and moving obstacle detection, and moving obstacle tracking. We have developed the framework for an efficient faces database that was used by the face recognition system for recognizing the customer.

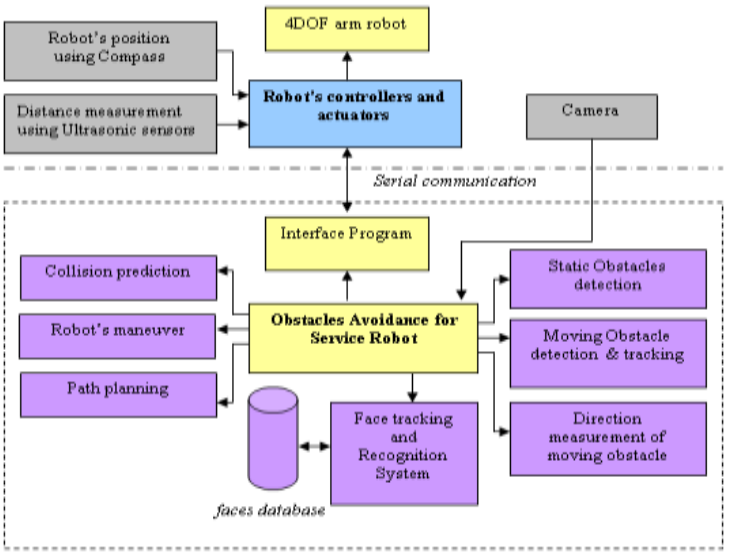


Figure 4 General architecture of service robot Srikandi II. The hardware and software parts are separated by a dashed line. All arrows indicate dataflow.

3 Proposed Method

3.1 General Obstacles Avoidance Method

The robot needs information from the ultrasonic sensors about the distance between the obstacle and the robot to be able to avoid a collision. Ultrasonic sensors generally obtain the radial measurement of the nearest distance from an object located in area A, as shown in Figure 5. Ultrasonic sensors work at a frequency of 40 KHz and have a maximum deviation angle of about 30°, so usually robots need more than one sensor to be able to measure the distance to an obstacle in its vicinity. The main weakness of this type of sensor is the interference between different sensors and the limited ability to detect the

obstacle. The advantage of this type of sensor is that it is usually able to detect the obstacle at a distance >=3 cm, something a vision sensor is not able to do.

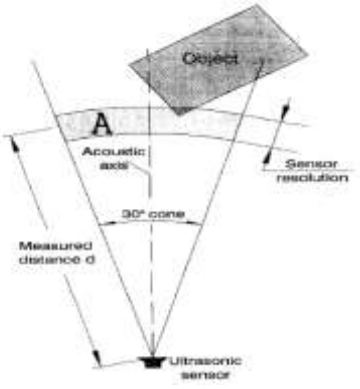


Figure 5 Two-dimensional projection of conical perception field of ultrasonic sensor. Distance measurement d indicates the presence of an object within the area [7].

Theoretically, using three ultrasonic sensors is enough to obtain distance information from the front, left and right sides of the robot. Therefore, we propose the obstacle avoidance model for service robots as shown in the flowchart in Figure 6. This method is a combination of static and moving obstacle detection, using vision and ultrasonic sensors.

Information about static obstacles is obtained as soon as the robot starts moving. Images from the landmark are captured and stored in a 640x480 JPEG file. The program will scan the images to find a free area that can be used by the robot. At the end of the program, the path from start to target position is calculated in order to guide the robot in its movement. The next step is object detection and face recognition for identification of the customer based on principal component analysis (PCA). Information about the moving obstacle is obtained when the robot detects and tracks someone or something moving in front it. The distance of the moving obstacle is obtained using distance sensors periodically, say every 1 seconds. Based on the information about the moving obstacle obtained by the vision sensor, we can estimate the speed of the moving obstacle, predict the collision point and determine the appropriate maneuvering action. Figure 6 shown below is a flowchart that describes the general mechanism for our method to detect and identify obstacles and make a maneuver to avoid collision:

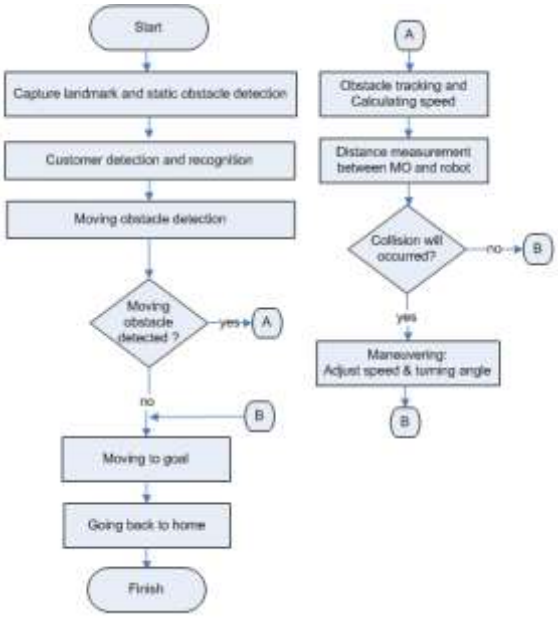


Figure 6 Flow chart of obstacle detection and avoidance action from start to target position for a vision-based service robot.

To implement the above flow chart in a service robot that should be able to recognize the customer and have the ability to avoid obstacles autonomously, we have developed algorithms and programs consisting of three main modules, namely a face recognition system, an obstacle detection system and an avoidance maneuvering method. The algorithm for the general obstacle avoidance method for a vision-based service robot moving from a specified starting point to a specified target position is as follows:

Algorithm 1. General method of obstacle avoidance by a service robot

```
Capture and saving landmark
Call staticObstacleDetection  // using Canny edge detector
Call faceDetectionRecognition  // Face recognition system
if (customer_identified ==true)
   do while not goal // while min_distance < distance measured by front sensor.
      set heading robot to the goal
      call movingObstacleDetectionTracking
      distance measurment
      robot running with normal speed
      if (moving obstacles==true) then
         calculate speed and Distance measurement of moving obstacle
       // if  minimum distance to maneuver reached , then maneuvering
        If ( d_safe  ==true )
           call maneuveringRobot
   endif
```

```
            move to goal position
        else
         Move to goal position  // if moving obstacle==false
        endif
      end while
      giving a cup to a customer
      move to home / start position // task finished
  else
       robot stop // if no customer detected
 endif
 function faceRecognition //More detailed in algorithm 2
         face detection and capture face customer
         training images
         testing face customer
         face recognition
 end function
function staticObstacleDetection
         captureLandmark and
         staticObstacleProcessing //More detailed in algorithm 3
 end function
 function movingObstacleDetectionTracking //More detailed in algorithm 3
          Moving obstacle detection
          Moving obstacle tracking
 End function
 function maneuveringRobot //More detailed in algorithm 4.
         calculate θ_m and v_R
         adjust turning angle and speed
         Update position of robot to the goal position
end function
```

3.2 Algorithm for Face Recognition System

We use PCA for the face recognition system to recognize the customer's face. Illumination and pose variations are mostly responsible for dramatic changes in the appearance of faces, which has proven to be a very difficult problem [16, 17]. The brightness level of face images in an indoor environment is a random variable. For solving the problems related to pose and illumination effects in indoor environments, we propose a method for training the system by varying the pose and illumination of training images using normal distribution generated by our program developed in C++, using the Technical Report Features Pack. Histogram equalization applied to the input images is a powerful method for automatically standardizing the brightness and contrast of facial images. We have developed a faces database called ITS database, and evaluated the success rate of our method using the Indian database [18] and the Yale University database [19]. The algorithm below shows the face recognition system for our service robot. Training images are assumed to be available beforehand, while testing images are obtained when the robot detects a face. When a customer is identified, the name of the customer is assigned to the variable *customer_name*.

Algorithm 2. Face recognition system for a service robot

```
// Function to store images as a faces database and train the faces.
Function captureAndTrainingImages
   capture face customer 640x480 pixel using 3 pose (front, left, right)
   do histogram equalization to face customer
    save to database
   training face images
end function

// Function to detects and recognizes a customer
 function faceDetectionRecognition
              face detection                    // Using Haar cascade classifier
           if  (face_detected ==true) then
            capture face customer 640x480 pixel
                face recognition          // testing face customer to faces database
                 if( (face_recognition==true) then
                      customer_identified=true
                      set  customer_name
                else
                     customer_identified=false; // no action for the robot if customer not identified
            endif
  end function
```

3.3 Algorithm for Obstacle Detection, Obstacle Tracking and Path Planning

The algorithm for fast static obstacle detection uses the Canny edge detector, moving obstacle tracking and path planning as shown in Algorithm 3 below; input landmark: 640x480 pixels in JPEG format. The calibration is necessary for establishing the threshold value for setting the pixels to white/black. White indicates a free area. Subsequently, the target position is obtained based on the largest free area. The predefined start and target position are used for path planning by the robot. Finally, the program will guide the robot to the target position using the compass sensor. When a moving obstacle is detected, it will be tracked and an estimation of the direction angle θ_O and velocity of the obstacle v_O will be calculated.

Algorithm 3. Fast obstacle detection and path planning

```
// Function to detect static obstacle using edge detection and path planning
function staticObstacleDetection
Capture and saving landmark
Canny edge detection and closing operation
while (not end of pixel from image(x,y) )
   if(avg. pixel >threshold)
              set pixel to  white
   else
        Set pixel to black
  endif
   end while
```

```
    Smoothing operation
    Display start and goal position  // draw a track from start to goal position
    Read position by compass sensor  // for heading the robot to goal position
    Set heading robot to goal position
end function

// Function to detects and tracks a moving obstacle
function movingObstacleDetectionTracking
        moving obstacle detection // Using Haar cascade classifier
          if (moving_obstacle==true) then
                distance measurement between  robot and moving obstacle
                //  tracking moving obstacle using motion history
                     different=frame_pre –frame_now // different of motion
                     Update motion history
                     Calculate motion gradient
                     θ_O = 360-Globalorientation(direction) // θ_O is obtained
                     Calculate
          endif
end function
```

3.4 Method and Algorithm for Avoidance Maneuvering

In this paper, we introduce a new geometric model for robots to maneuver in order to avoid a moving obstacle. Three ultrasonic sensors, at the front, left and right sides of the robot, continuously measure the distance between the robot and the moving obstacle to ensure that the distance between the robot and the obstacle remains larger than the minimum distance in order to avoid collision. The left and right sensors also make sure that if the moving obstacle comes from the left or right side of the robot it will be detected by these sensors. The proposed model to make an avoidance maneuver is based on the model shown in Figure 7.

From Figure 7, in which the constant of the collision angle is denoted as k_θ and the minimum distance to start maneuvering in order to avoid collision as d_{safe}: when the robot detects that the minimum distance to start maneuvering is reached, it will use turning angle θ_m if the robot is at the right/left side of the moving obstacle:

$$\theta_m = \begin{cases} \theta_R + (\frac{\pi}{2}(k_\theta / d_O)) \text{ if } \mathrm{d_O} < \mathrm{d_{safe}}, \text{robot at the right side of obs.} \\ \theta_R - (\frac{\pi}{2}(k_\theta / d_O)) \text{ if } d_O < d_{safe}, \text{robot at the left side of obs.} \end{cases} \quad (5)$$

If d_n, the minimum distance to redirect the robot to the target position, is reached, the turning angle is:

$$\theta_m = \theta_R \text{ if } d_{safe} > d_O > d_n \quad (6)$$

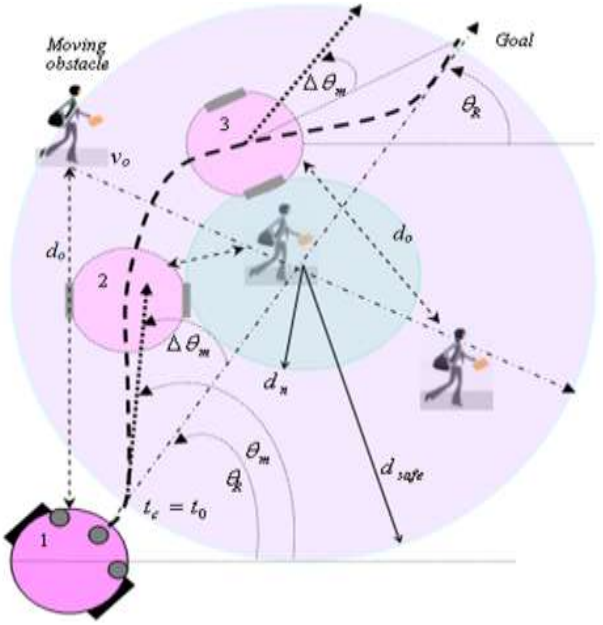


Figure 7 Model of a maneuvering method to avoid a moving obstacle. This shows a robot maneuvering smoothly to avoid collision with a moving object based on a calculation of the distance between the robot and the moving obstacle.

If $d_{safe} > d_O > d_n$ is reached, the robot will start to redirect itself to the target using its compass sensor, based on previously obtained data. The velocity of the robot v_R when in avoidance mode is computed between $\frac{v_{\max}}{2}$ – $v_{\max}$; the maximum speed occurs when $\Delta\theta_m$ is $\frac{\pi}{2}$, for which the proposed formula is:

$$v_R = v_{\max}\left(1 - \left|0.5 - \frac{\Delta\theta_m}{\pi}\right|\right) \quad (7)$$

Based on the geometric model shown in Figure 6, the algorithm for the avoidance maneuvering method is:

Algorithm 4. Avoidance maneuvering method for robot.

// Function to maneuver the robot to avoid collision with a moving obstacle
function maneuveringRobot

if $d_O < d_{safe}$ **then**
set maneuver=true
calculate θ_m and v_R for maneuver
adjust turning angle and speed

endif
if $d_{safe} > d_O > d_n$ **then**
set maneuver==false
set $\theta_m = v_R$ for reorientation to the goal
adjust turning angle and speed

endif
Update position of robot to the goal position
end function

4 Experimental Results

For conducting obstacle avoidance experiments, a self-navigating service robot has been programmed to deliver a cup in our 4 x 5 meter lab as a simulation of an indoor environment. Face tracking and face recognition based on eigenspaces with three images per person were used. Because of the limitations of the space, people as moving obstacles only walked straight in front of the robot and approached the robot directly. The normal speed of the robot was 0.2 m/s but varied when the robot maneuvered. We defined a minimal distance between robot and obstacle, *min_distance* = 40 cm, for the robot to start maneuvering as an emergency response to avoid collision with the obstacle. We set up the experiment so that a moving obstacle would collide with the robot using the parameters shown in Table 1:

Table 1 Variables used in our experiments.

Variables	Values
θ_O	0°-80°
v_O	0.2-0.8 m/s
v_R	0.2-0.4m/s
T	1s
θ_{Cam}	30°
d_{safe}	120 cm
d_n	80cm
k_θ	60
min_distance	40 cm

Figure 8 shown below, is the set-up of the experiment carried out for this research. We used a chair and a table as static obstacles and someone walking straight in front of the robot, approaching it as a moving obstacle.

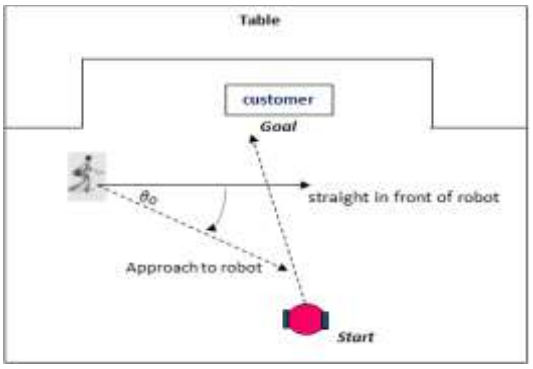


Figure 8 Set-up of experiment for detecting static and moving obstacles, navigating and maneuvering robot from start to target.

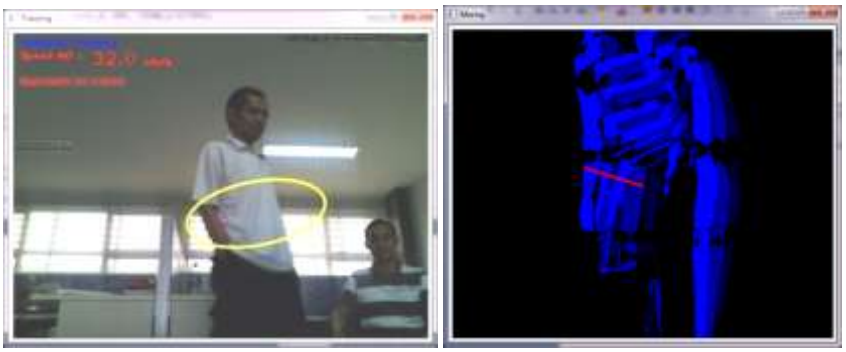
The procedure of the experiment was as follows. First, the robot captures the landmark and processes the images to determine static obstacles and the start-target positions. Then the robot detects if the customer is there or not; if the customer is there, the program will detect and track moving obstacles. Figure 9, shown below, indicates that our method for detecting static obstacles worked well in our experiment and our program was able to determine the free path from the start position to the target position. We used the Canny edge detector's closing operation and smoothing to reduce noise from the images. We used the upper body feature for moving obstacle detection and the frontal face feature from OpenCV library [20].

(a) (b)

Figure 9 Result of service robot simulation for detecting static obstacles and path planning from start to target position. Straight line from start to goal as guidance for the robot to move (a). Edge detecting and image processing for finding static obstacles. The white area indicates space with no obstacles (b).

The next step is customer detection and recognition. We evaluated the results of the proposed face recognition system using the Indian and Yale University faces databases. During our experiment, the ITS and Yale University faces databases had a higher success rate than the Indian faces database when we varied or did not vary illumination [18]. When a moving obstacle appeared and collision threatened, the robot would maneuver to avoid the obstacle, turning angle and speed being based on Eqs. (5)-(7). For establishing the direction of the moving obstacle, we used a tracking and motion history method as shown in Figure 10.

(a)



(a) (b)

Figure 10 Improved face recognition system to identify the customer (a). Detecting and tracking a moving obstacle and information about the speed of the moving obstacle (b). Using motion history, with the direction of the movement along the red line indicating that a person approaches the robot (c). Upper body library is used for detecting moving obstacles.

Figure 11, shown below, is the general result of the experiment. It shows clearly that the robot avoided the obstacle and moved to the target safely with the proposed avoidance maneuvering method.

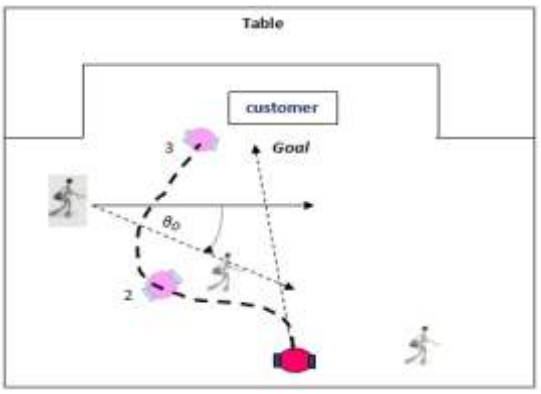
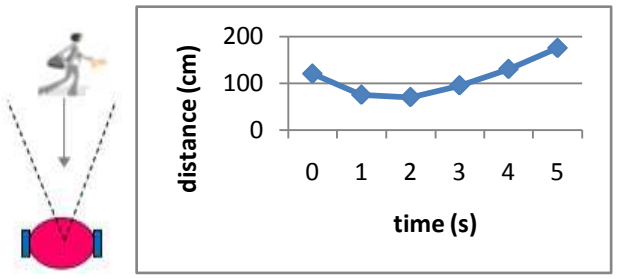
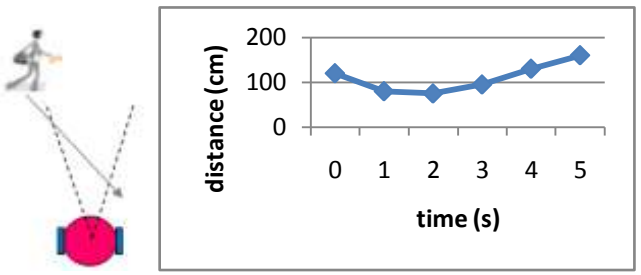


Figure 11 Result of experiment with obstacle avoidance by a service robot in an indoor environment. It shows that the robot successfully avoided the static and moving obstacles and reached the target position.

To identify the obstacle avoidance behavior using the proposed method, we examined our method on the basis of some quantitative measurements for different scenarios. We wanted to know the behavior of our method if the direction of the moving obstacle varied; see Figure 12.

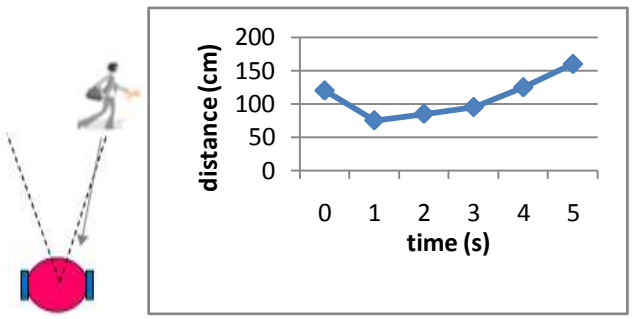


(a)



(b)

Figure 12 Result of experiment using different scenarios (a), (b) and (c) to obtain obstacle avoidance behavior using our method. The speed of the obstacle v_o is 40 cm/s, while the speed of the robot v_R is 20 cm/s.



(c)

Figure 12 ***Continued.*** Result of experiment using different scenarios (a), (b) and (c) to obtain obstacle avoidance behavior using our method. The speed of the obstacle v_o is 40 cm/s, while the speed of the robot v_R is 20 cm/s.

5 Discussion of Results

In general, the proposed method for obstacle avoidance has been successfully implemented and it has shown a good performance. Three ultrasonic sensors succeeded to detect and measure the distance of a moving obstacle continuously. Despite noise or imperfect distance measurements, it still gave the expected action and the robot maneuvered smoothly. The adjustments of the maneuvering angle and speed of the robot were very smooth because in the proposed formula they depend on the distance to the moving obstacle proportionally.

There was no oscillation when the robot traveled to the target because the robot uses path planning and was guided by a compass. The robot was able to identify the customer because the face recognition system had enough training images to adapt to pose and illumination variations. An indication of the speed of performance of our method and algorithms is sampling time S (i.e. the speed at which the steer and speed commands by the low-cost controller were issued. The following events occur during S :

1. Obtain sonar information from the sensor controller
2. Calculate the moving obstacle distance
3. Process the information about the moving obstacle from the vision sensor
4. Calculate the speed command and determine the action
5. Communicate with the motion controller in order to send speed and steer commands.

An Intel Atom 1.6 GHz and 1 GB RAM laptop S =220 ms was used, making the proposed system for obstacle avoidance by service robots fast and reliable.

Table 2 provides a comparison between our method and common methods PFM and VFH, showing that our method has the important additional feature that it can be used with low-cost sensors:

Table 2 Comparison between PFM, VFH and our method.

Variables	PFM	VFH	Our method
Provide a complete mechanism for a vision-based service robot	x	x	√
Low-cost sensors	√	x	√
Fast obstacle avoidance	√	√	√
No oscillations in the presence of obstacles	x	√	√

6 Conclusions

This paper presents a new obstacle avoidance method for service robots in indoor environments. Algorithms for obstacle avoidance and a geometric model for making an avoidance maneuver were implemented in a service robot. Integration of vision and ultrasonic sensors proved applicable as main sensors for the service robot. Experimental results with various scenarios have shown that the robot reached the target point while avoiding moving obstacles with the proposed method, and was not limited by an increase in complexity of the system as the environment changed. The obstacle avoidance method proposed has shown a good performance and could be an important feature, especially for vision-based service robots. There is no oscillation when the robot travels to the target because the robot uses path planning and is guided by a compass. The sensor system is very cheap because it only uses three distance sensors. From quantitative measurements it followed that obstacle avoidance based on our method worked very well for different scenarios. In the future, we will model a navigation system for a vision-based service robot with the ability to detect multiple moving obstacles and additional state estimation using the Bayesian approach.

References

[1] Acosta, L., González, E.J., Rodríguez, J.N., Hamilton, A.F., Méndez, J.A., Hernéndez, S., Sigut, S.M. & Marichal, G.N., *Design and Implementation of a Service Robot for A Restaurant*, International Journal of Robotics and Automation, **21**(4), pp. 273-281, 2006.

[2] Qing-wiau, Y., Can, Y., Zhuang, F. & Yan-Zheng, Z., *Research of the Localization of Restaurant Service Robot*, International Journal of Advanced Robotic Systems, **7**(3), pp. 227-238, 2010.

[3] Borenstein, J. & Koren, Y., *The Vector Field Histogram - Fast Obstacle Avoidance for Mobile Robots*, in Proc. IEEE Trans. On Robotics and Automation, **7**(3), pp. 278-288, 1991.

[4] Kahraman, F., Kurt, B. & Gokmen, M., *Robust Face Alignment for Illumination and Pose Invariant Face Recognition*, In-Tech publishing, pp.239-240, 2010.

[5] Masehian, E. & Katebi, Y., *Robot Motion Planning in Dynamic Environments with Moving Obstacles and Target*, Int. Journal of Mechanical Systems Science and Engineering, **1**(1), pp. 20-25, 2007.

[6] Minura, J., Uozumin, H. & Shirai, Y., *Mobile Robot Motion Planning considering the Motion Uncertainty of Moving Obstacles*, in Proc. IEEE Int. Conf. on System, Man, and Cybernetics, pp. 692-698, 1999.

[7] Khatib, O., *Real-time Obstacle Avoidance for Manipulator and Mobile Robots*, International Journal of Robotics Research, **5**(1), pp. 90-98, 1986.

[8] Borenstein, J. & Koren, Y., *Potential Field Methods and Their Inherent Limitations for Mobile Robot Navigation*, in proc. IEEE Conf. on Robotics and Automation, California, pp.1398-1404, 1991.

[9] Lidoris, G., Wollherr, D. & Buss, M., *Bayesian State Estimation and* Behavior *Selection for Autonomous Robot Exploration in Dynamic Environments*, IEEE International Conf. on Intelligent Robots and Systems, France, 2008.

[10] Fulgenzi, C., Spalanzani, A. & Laugier, C., *Dynamic Obstacle Avoidance in Uncertain Environment Combining PVOs and Occupancy Grid*, IEEE International Conf. on Robotics and Automation, Italy, 2007.

[11] Thrun, S., Probabilistic *Robotics*, The MIT Press, pp. 10-30, 2006.

[12] Li, Y. & He, K., *A Novel Obstacle Avoidance and Navigation Method of* Outdoor *Mobile Robot*, in Proc. 12th International Conf. on Advanced Robotics, pp. 653-656, 2005.

[13] Foka, A. & Trahanias, E., *Predictive Control of Robot Velocity to Avoid Obstacles in Dynamic Environments*, IEEE International Conf. on Intelligent Robots and Systems, Nevada, pp. 370-375, 2003.

[14] Budiharto, W., Purwanto, D. & Jazidie, A., *Indoor Navigation using ANFIS* controller *for Servant Robot*, in Proc. IEEE 2nd International Conference on Computer Engineering and Its Application (ICCEA 2010), Bali-Indonesia, pp. 582-586, 2010. DOI: 10.1109/ICCEA.2010.119.

[15] Budiharto, W., Purwanto, D. & Jazidie, A., *A Novel Method for Static and Moving Obstacle for Service robot using Bayesian Filtering*, IEEE 2nd International Conf. on Advances in Computing, Control and Telecomunications Technology, 2-3 December, Jakarta, 2010, DOI: 10.1109/ACT.2010.51.

[16] Turk, M. & Pentland, A., *Face Recognition Using Eigenfaces*, In Proceeding of IEEE Conference on Computer Vision and Pattern Recognition, pp. 586-591, 1991.

[17] Yang, M., *Detecting Faces Images: A Survey,* IEEE Transactions on Pattern Analysis and Machine Intelligence, **24**(1), pp. 34-58, 2002.
[18] Indian Face Database, http://vis-www.cs.umass.edu/~vidit/IndianFace Database, last accessed 21 February 2010.
[19] ATT Face Database, http://www.cl.cam.ac.uk/research/dtg/attarchive/ facedatabase.html, last accessed 10 March 2010.
[20] OpenCV, http://sourceforge.net/projects/opencvlibrary, last accessed on 10 November 2011.